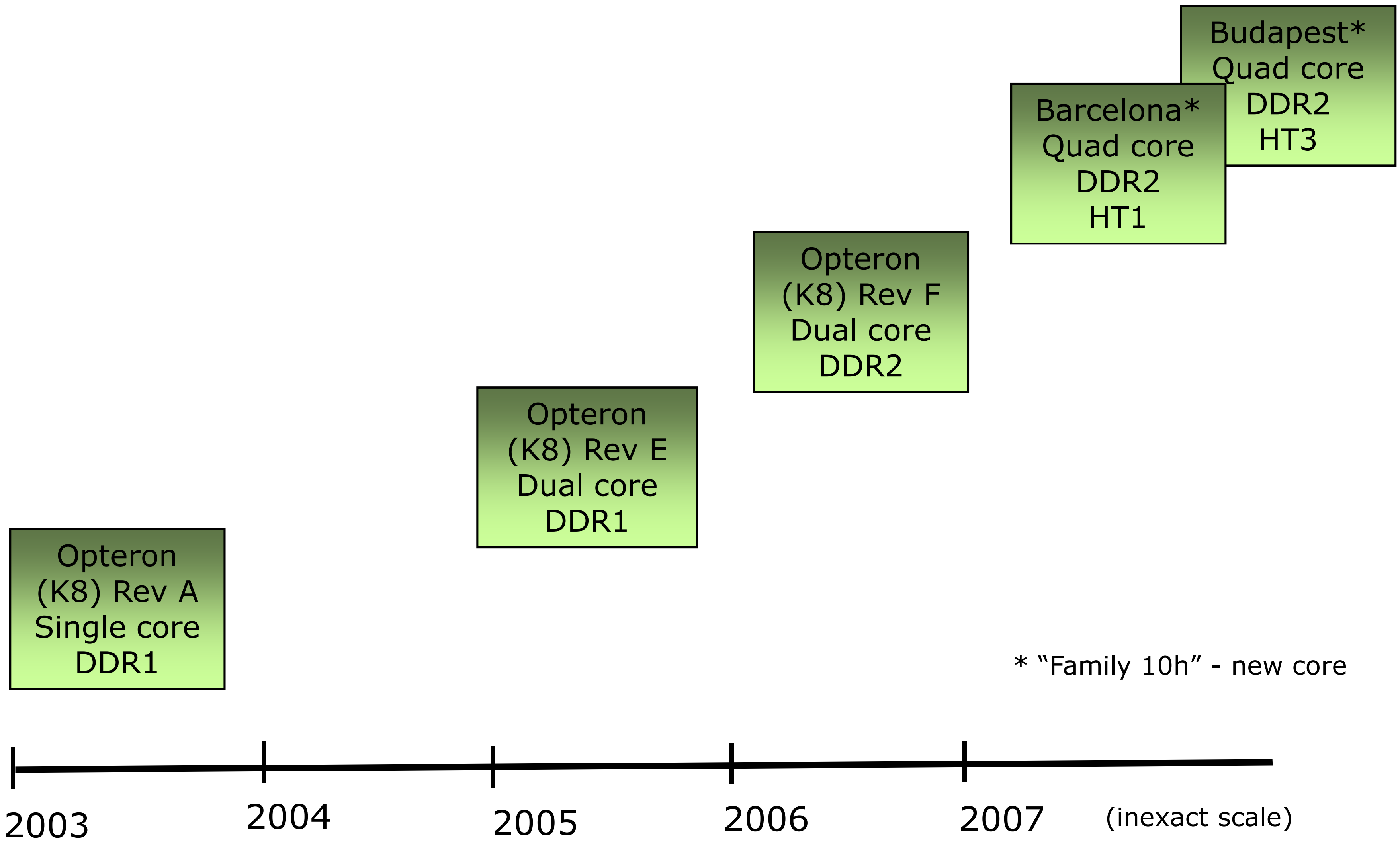
AMD Quad Core Processor Overview

Brian Waldecker, Ph.D.
Senior Member of Technical Staff
AMD, Austin
7/30/2007

Outline

1. Background and First Dual Core Opteron Experiences
2. Multi-Core Processor Architecture
3. FPU Enhancements
4. Core IPC Enhancements
5. Cache Hierarchy and Structure
6. TLBs and Large Pages
7. NB Enhancements
8. Prefetching
9. (Some) Recommended Programming Practices

AMD Multi-Core Evolution

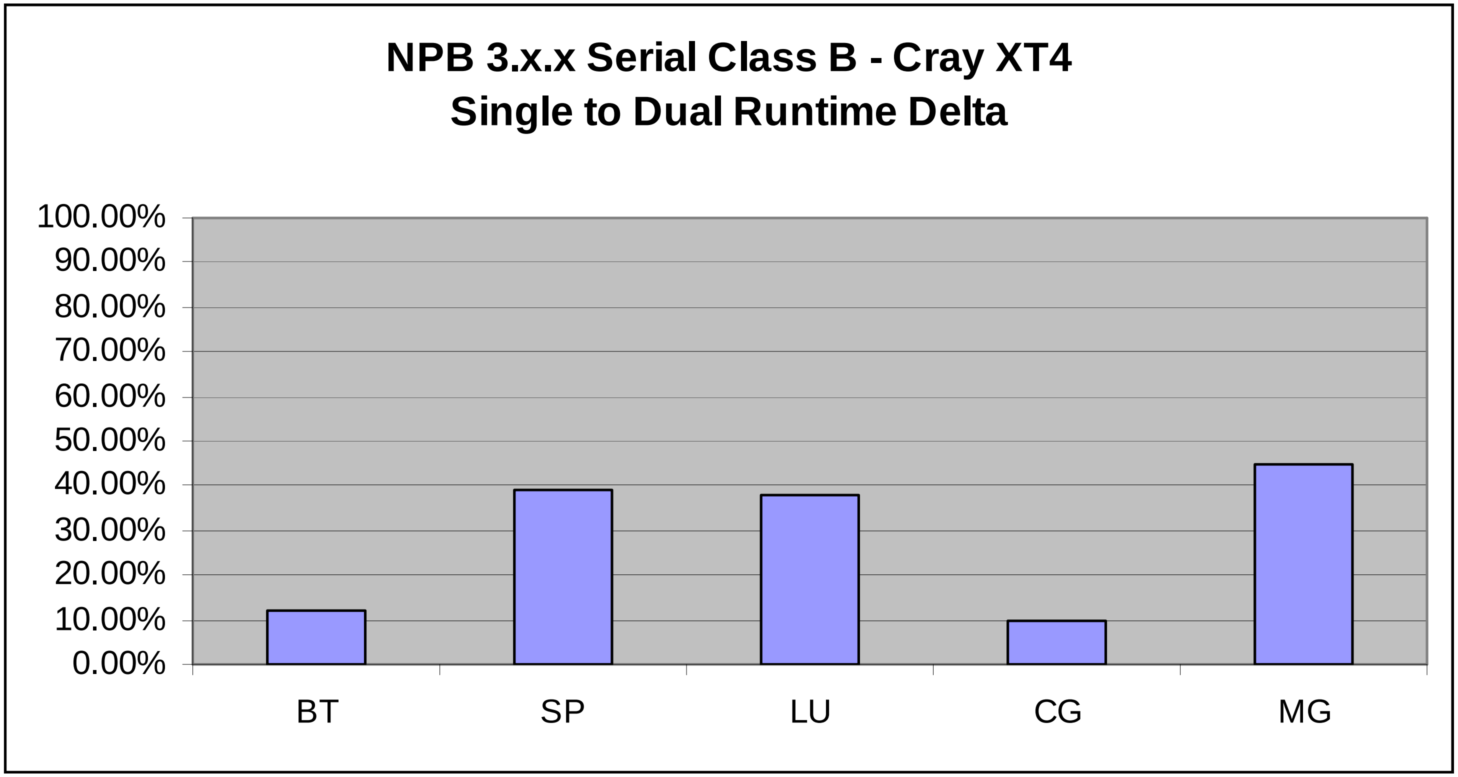


Results from recent history
NPB Serial Single to Dual Core

Cray XT4 Internal Machine
2.6GHz Opteron RevF cpus
DDR2-667 memory
Small Pages

Methodology:
1. Run one 1 copy on 1 core (of 2)
2. Run a copy on both cores concurrently
3. Compare per copy completion times

(note: 2x work done in 2 copies/chip case)



More Results
Single to Dual Core for MILC & GAMESS

Methodology: (XT3 had dual core RevE cpus, XT4 had dual core RevF cpus).
1. Number of MPI Ranks held constant.
2. Run one 1 MPI rank per chip (leave 2nd core idle).
3. Run 2 MPI ranks per chip.
4. Compare wall clock times.

note: 2 ranks/chip (dual core) used half as many chips.

Increase in Wall Clock Time (%) for Single Core to Dual Core

Application	XT3	XT4
MILC (quantum chromodynamics)	44%	43%
GTC (plasma turbulence, particle in cell method)	4%	
Paratec (ab initio quantum mechanical)	5%	4%
CAM (atmospheric modeling)	12%	10%
MADbench (cosmic microwave background)	1%	2%
GAMESS (ab initio quantum chemistry)	2%	

Semantic Note: An increase of 50% in Wall Clock Time means new runtime was 1.5X the old runtime.

What's Next for AMD?

Mid-2007 — Quad-Core AMD Opteron™ Processors

More than just four cores

- Significant CPU Core Enhancements
- Significant Cache Enhancements

World-class performance goals

Native Quad-Core
- Faster data sharing between cores

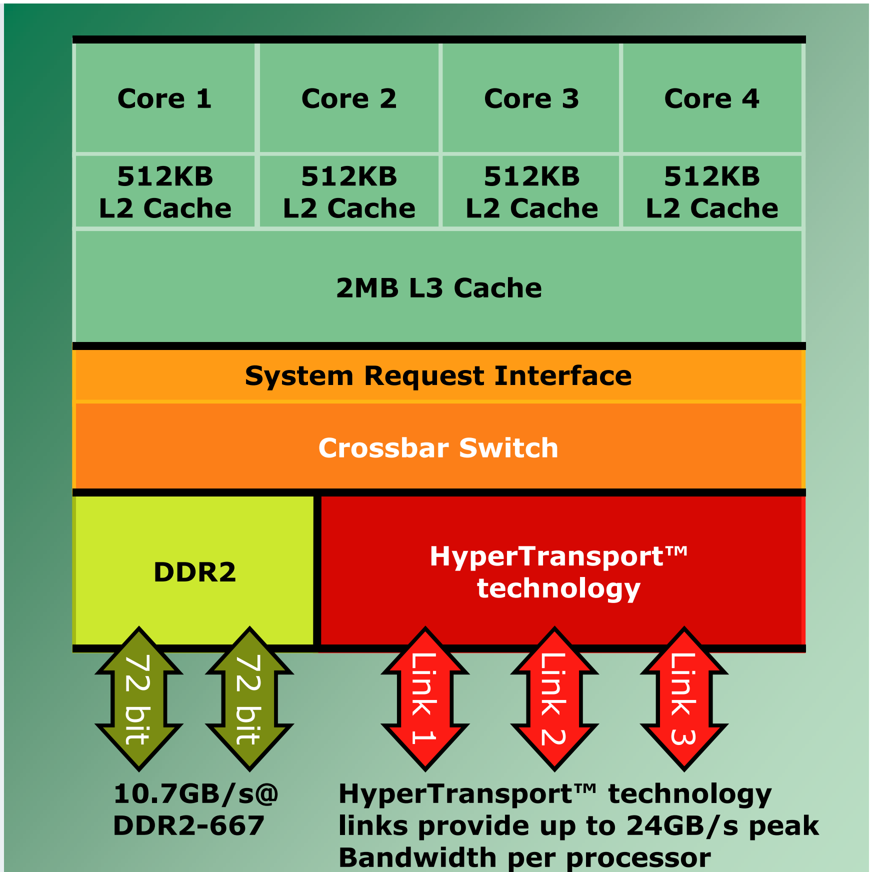
AMD Virtualization™ enhancements
- Nested paging acceleration for virtual environments

Reducing total cost of ownership

Performance/Watt leadership
- Consistent 95W thermal design point
- Low power 68W solutions

Drop-in upgrade
- Socket F compatibility – BIOS upgrade
- Leverage existing platform infrastructure

Common Core Architecture
- One core technology top-to-bottom
- Top-to-bottom platform feature consistency



Quad-Core AMD Opteron™ Processor Design for Socket F (1207)

AMD Quad-Core Processor Architecture
A Closer Look at Barcelona

AMD Smarter Choice

Comprehensive Upgrades for SSE128

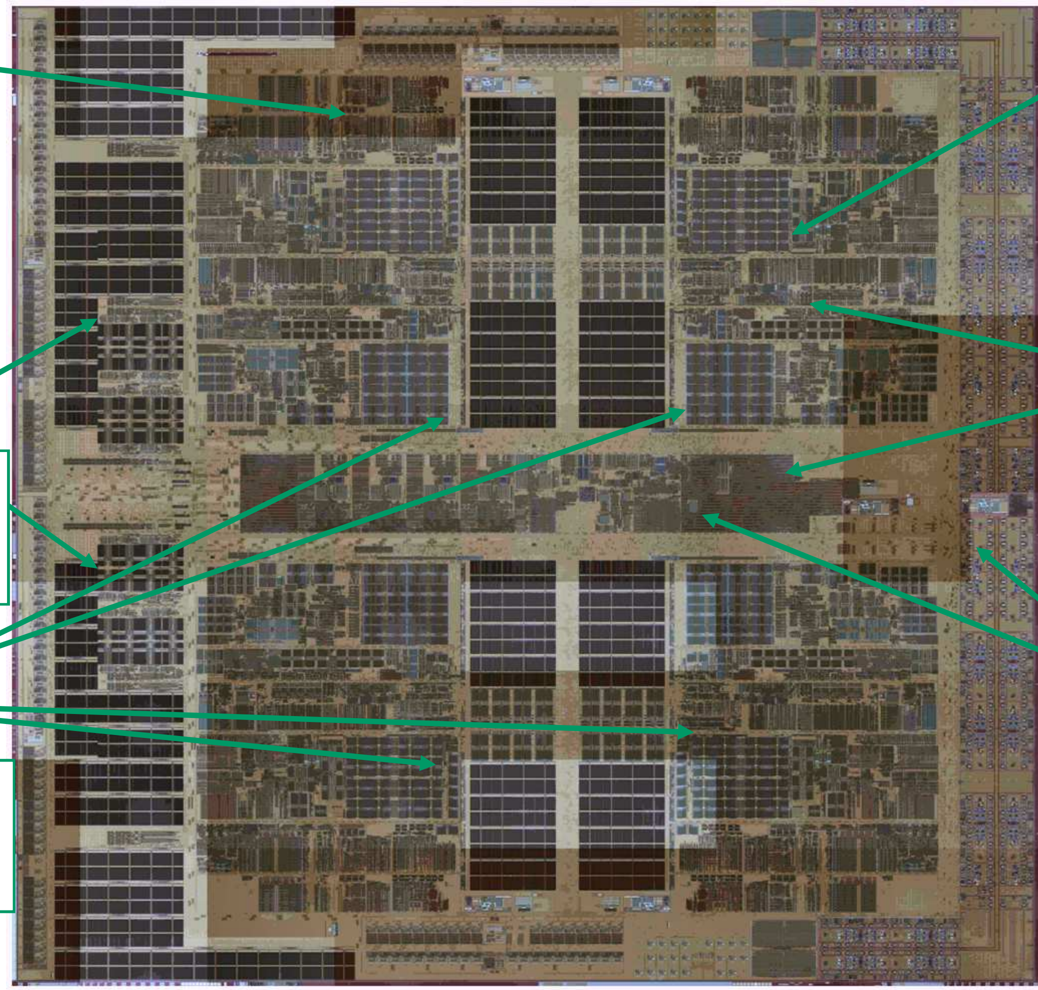
Quadruples floating-point capabilities

New Highly Efficient Cache Structure including a shared L3

Balance of dedicated and shared cache for optimal Quad-Core performance

Enhanced CPU Cores

Benefits all applications by improving the overall efficiency and performance of the cores



Enhanced Virtualization

New "Nested Paging" feature designed for near native performance on virtualization applications

Advanced Power Management

Provides granular power management resulting in improved power efficiency

DRAM Controller Enhancements

Specifically tuned for Quad-Core memory accesses, improves overall memory performance

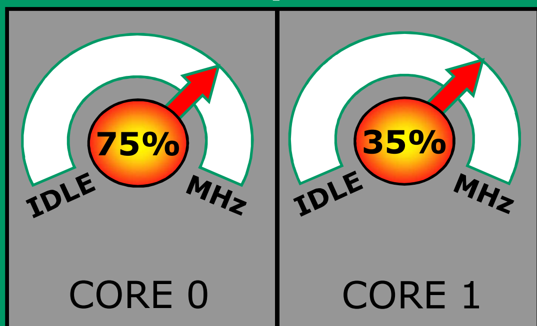
Improved Processor Power Management

with Enhanced AMD PowerNow!™ Technology

AMD
Smarter Choice

"GOOD"

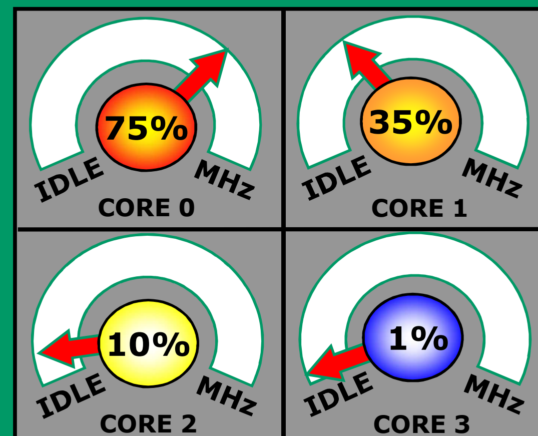
Current AMD Opteron™ Dual-core processors



MHz and voltage is locked to highest utilized core's p-state

"GREAT"

'Barcelona'



MHz is independently adjusted separately per core*. Voltage is locked to highest utilized core's p-state

* programmer note: separate TSC per core; increments at rate specified in an MSR, not just core MHz

Native Quad-Core technology enables enhanced power management across all four cores

Comprehensive Upgrades for SSE128

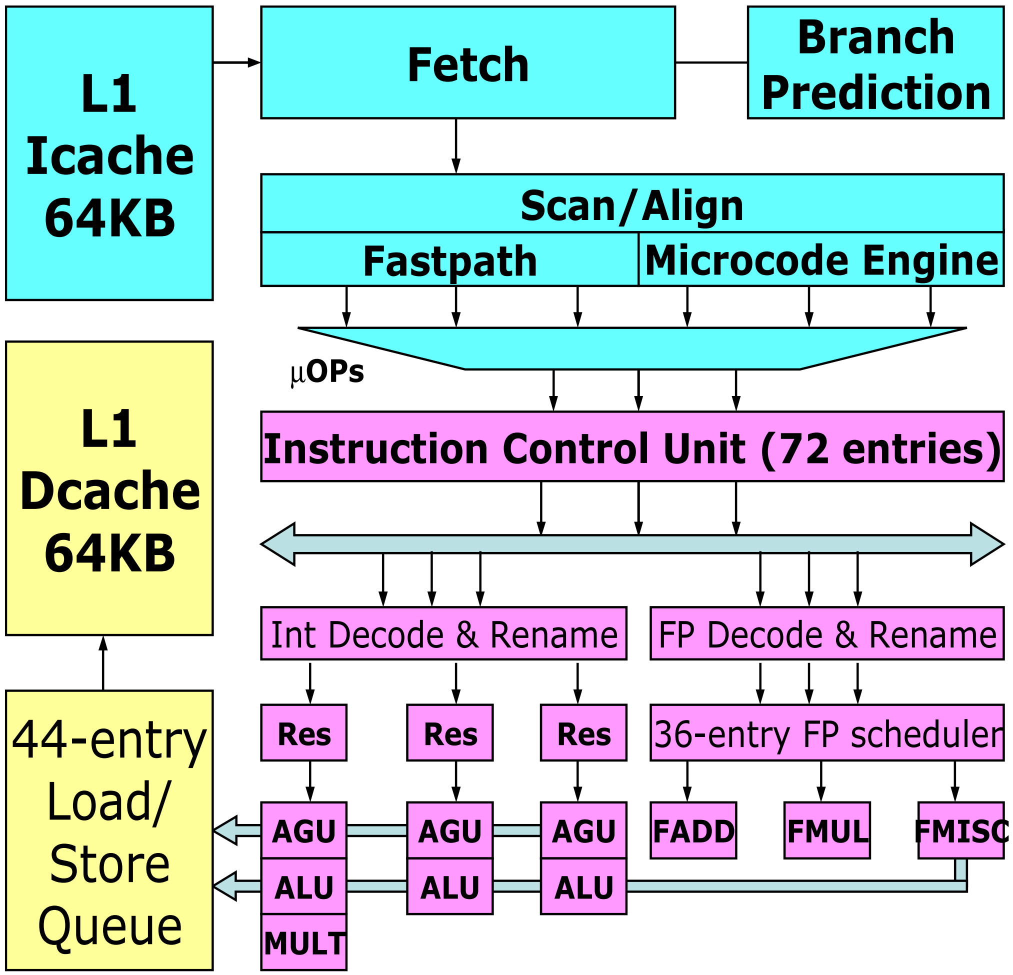
128bit FPU

Parameter	Current Processor	"Barcelona"
SSE Exec Width	64	128 + SSE MOVs
Instruction Fetch Bandwidth	16 bytes/cycle	32 bytes/cycle + unaligned Ld-Ops
Data Cache Bandwidth	2 x 64bit loads/cycle	2 x 128bit loads/cycle
L2/NB Bandwidth	64 bits/cycle	128 bits/cycle
FP Scheduler Depth	36 Dedicated x 64-bit ops	36 Dedicated x 128-bit ops

- Can perform SSE MOVs in the FP "store" pipe
  - Execute two generic SSE ops + SSE MOV each cycle (+ two 128-bit SSE loads)
- SSE Unaligned Load-Execute mode
  - Remove alignment requirements for SSE ld-op instructions
  - Eliminate awkward pairs of separate load and compute instructions
  - *To improve instruction packing and decoding efficiency*

Core IPC improvements

- Improve Branch Prediction.
- TLB enhancements.
- More out of order Ld/St capability.
  - Loads can bypass other loads and non-conflicting stores.
  - LS1 queue (12 entries) - can issue 2 operations per cycle (load or store tag check).
  - LS2 queue (32 entries) - holds requests that miss in L1 cache.
- New Instructions
  - (ABM) POPCNT / LZCNT
  - (SSE4A) EXTRQ / INSERTQ / MOVNTSD / MOVNTSS
  - (SSE3) MONITOR/MWAIT
- Fastpath support for FP to Integer data movement.



Cache Hierarchy

AMD Smarter Choice

Dedicated L1 cache

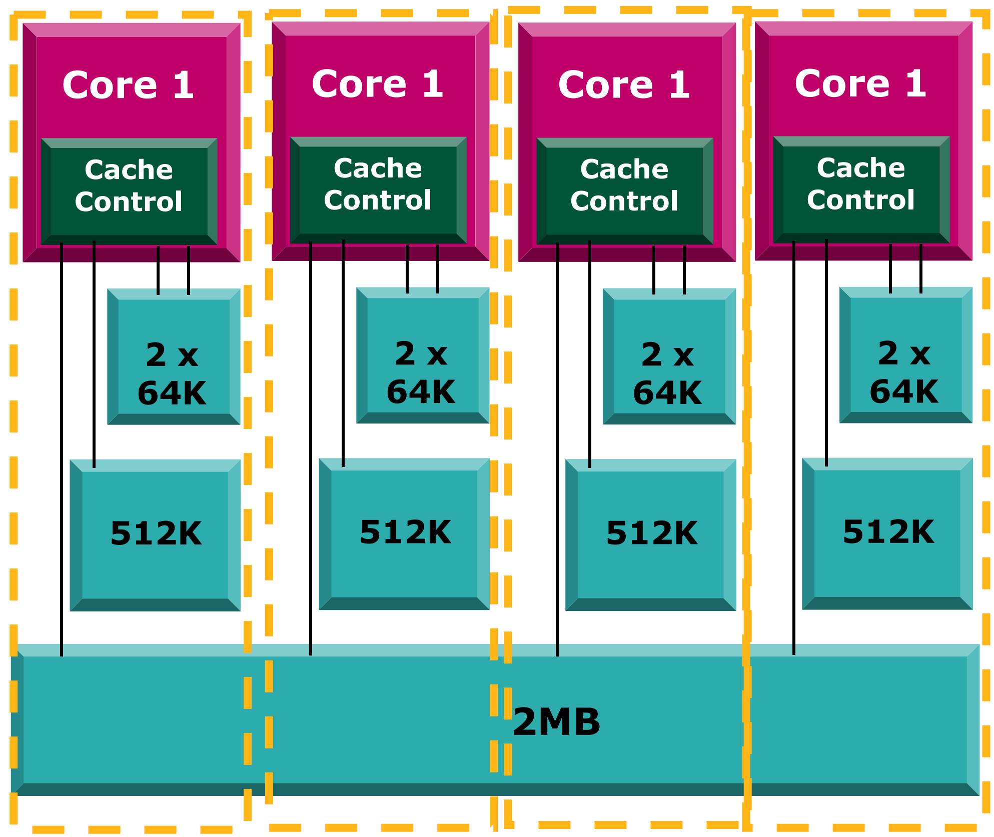
- 2 way associativity.
- 8 banks (each 16B wide).
- 2 128bit loads/cycle, 2 64 stores/cycle, or combination
- 3 cycle load to use delay.

Dedicated L2 cache

- 16 way associativity.
- victim cache, exclusive w.r.t L1
- ~12 cycle latency.

Shared L3 cache

- 32 way associativity.
- victim cache, partially exclusive w.r.t. L2
- fills from L3 leave likely shared lines in L3.
- sharing aware replacement policy.



replacement policies
L1, L2: pseudo LRU
L3: sharing aware pseudo LRU

TLB Enhancements

- Support for 1GB pagesize (4k, 2M, 1G)
- 48 bit physical addresses = 256TB (increase from previous 40bits)
- Data TLB
  - L1 Data TLB
    - 48 entries, fully associative
    - all 48 entries support any pagesize
  - L2 TLB
    - 512 4k entries, or
    - 128 2M entries

Instruction TLB

- L1 Instruction TLB
  - fully associative
  - support for 4k or 2M pagesizes
- L2 Instruction TLB

Memory Controller Enhancements

Independent (unganged mode) DRAM controllers

- allow more concurrent reads of needed data.
- reduce bank conflicts.
- longer burst length = better command efficiency.

Optimized DRAM paging

- adaptive closing of DRAM pages = more page hits.

Re-architect Northbridge for higher BW

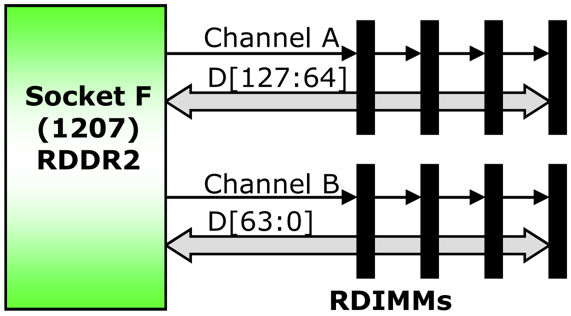
- resize buffers, better command scheduling, DDR2 and beyond.

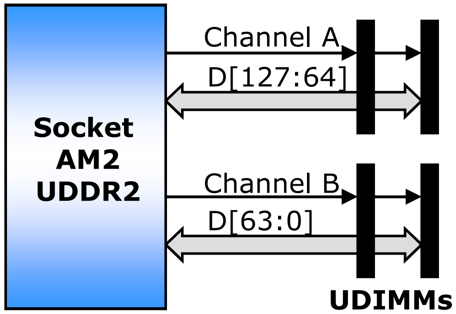
Write bursting

- reduce read-write turnarounds.

Memory Controller Configurations

- Supports DDR2
  - Two DDR2 channels
  - Target speeds up to DDR800 for DDR2
  - Up to eight registered DDR2 DIMMs with Socket F (1207), DRAM speed may be limited
  - Up to four unbuffered DDR2 DIMMs with Socket AM2
  - 2T timing provided for >1 UDIMM per channel due to address/command loading conditions
  - Adaptive closing of DRAM pages for fewer page conflicts.
- Channel configurability:
  - Two DCTs ganged together in 128-bit mode (DDR2)
  - Two DCTs unganged for two independent 64-bit DRAM controllers (DDR2)





Ganged vs. Unganged Memory Channels

Ganged channels (DDR2)

- DCT channels A and B can be ganged as a single logical 128-bit DIMM
- Offers highest DDR2 bandwidth
- Requires both DIMMs in a logical pair to have identical size and timing parameters, both DCTs programmed identically

Unganged channels

- DCT channels A and B operate as two completely independent 64-bit channels (both channels operate at the same frequency)
- Reduce dram page conflicts – more concurrent open dram pages
- Better bus efficiency

Burst lengths supported

- 128-bit ganged operation supports burst length of four only with DDR2

  128-bits x 4=64 bytes
- 64-bit (unganged) operation supports either burst of four or eight with DDR2

  64-bits x 4 = 32 bytes

  64-bits x 8 = 64 bytes (provides highest bus utilization)

Memory Controller - Write Bursting

DRAM writes can be buffered in the memory controller (MCT) before being bursted to the DRAM controller (DCT) to improve DRAM interface efficiency

- BIOS programmable from 2-32 writes, disabled at reset
- Applies only to low-priority writes, medium and high priority writes are not stalled for write bursting
- Once the programmable threshold is reached, all writes in the memory controller queue are converted to medium priority
  - *Address matched writes are promoted to medium priority or the priority of the subsequent access, whichever is higher*
- Some low-priority writes may be sent to the DCT prior to reaching the burst threshold
  - *BIOS programmable, 0, 1, 2, or no limit*
- Writes can be flushed by BIOS or optionally on Stop Grants and periodically based on scrub rate

Data Prefetch- Hardware Prefetchers

Hardware prefetching

- DRAM prefetcher
  - tracks positive, negative, non-unit strides.
  - dedicated buffer (in NB) to hold prefetched data.
  - Aggressively use idle DRAM cycles.
- Core prefetchers
  - Does hardware prefetching into L1 Dcache.
  - Improvements over K8
    - Adaptive prefetching – increases prefetch distance if demand stream catches prefetch stream.
    - Detection of cacheline pattern L, L+1 will trigger next N lines to be prefetched (N programmable unlike K8).
    - No hole as in K8. Lines L, L+1 cause prefetching of L+2, L+3, L+4 unlike K8 which would prefetch L+3, L+4 (leaving hole at L+2 to be filled by a demand miss).

Data Prefetch- Software Prefetch

Software prefetching instructions

- MOV (prefetch via load / store)
- prefetcht0, prefetcht1, prefetcht2 (currently all treated the same)
- prefetchw = prefetch with intent to modify
- prefetchnta = prefetch non-temporal (mark as LRU, favor for replacement)

Programming Hints - Prefetching

Which Prefetch to use ?

Data	Less than ½ L1 size	Less than ½ L2 size or of unknown size		Greater than ½ L2 size
		Reused	Not Reused	
Read only	`prefetch` or `prefetchnta`	`prefetch`	`prefetchnta`	`prefetchnta`
Sequential read only	`hwprefetcher + prefetch`	`hwprefetcher + prefetch`	`prefetchnta`	`prefetchnta`
Read-write	`prefetchw`	`prefetchw`	`prefetchnta`	`prefetchnta`
Sequential read-write	`prefetchw`	`prefetchw`	`prefetchnta`	`prefetchnta`
Write only	`prefetchw`	`prefetchw`	`movnt`	`movnt`
Sequential write only	`hwprefetcher + prefetchw`	`hwprefetcher + prefetchw`	`movnt`	`movnt`

Programming Hints - Prefetching

- Generally good to prefetch 6 to 8 cachelines ahead

- Try to have 100 cycles of computation in loop body between successive prefetches

- Avoid issuing multiple software prefetches to the same cacheline

- Avoid software prefetch to addresses within 64B of a store instruction (can create false dependency, stalling prefetch)

- Unroll loops enough times so each iteration works on 1 or more cachelines of data.

note: neither hw or sw prefetches will be allowed to generate page faults, but a TLB miss on a prefetch can initiate a TLB fill.

Programming Hints
128bit XMM registers

• In K8, the 128 bit XMM registers were implemented as 2 64 bit registers (upper and lower halves).

• For Barcelona, XMM registers are true 128 bit registers to match wider 128 bit FPUs.

• Instructions modifying only upper or lower half of a XMM register now must *merge* modified and unmodified halves.

• Bottom Line: It is preferable to use vector instructions that modify the entire XMM register to avoid merge overhead.

– (note: Instructions clearing one half and storing to the other qualify as modifying entire register and incur no penalty).

– See Software Optimization Guide for Family “10h” processors for further discussion and list of instructions.

Programming Hints
Cache Banks and Alignment

- L1D cache *bank conflicts* will reduce loads hitting in L1 from 2 per cycle to 1.
  - L1D cache is implemented as 8 banks. (Software optimization guide gives more detail).
  - On K8, if two addresses differ in bits 14:6 but are the same in bits 5:3, a L1D bank conflict is possible (but not definite).
  - On Barcelona, bits 5:3 become bits 6:4 (misaligned boundary now 16 bytes)
  - Conflict never occurs for walking a single array.
  - Conflict is *possible* for certain interleaved access patterns to multiple arrays.

- On Barcelona, ~50% fewer misaligned accesses and 50% improvement to misaligned bandwidth.

Summary

Barcelona pays attention throughout to improving:

- Core IPC.
- Caches and TLB.
- FPU performance.
- Memory performance.
- Compatibility, Usability, and Programmability.
- Support of future acceleration innovation.

References

- Opteron Processor Families Technical Documentation
  - http://www.amd.com/us-en/Processors/TechnicalResources/0,,30_182_739_9003,00.html
- *Bios and Kernel Developers Guide* (BKDG)
  - RevF cpus are denoted as "Family 0Fh".
  - Barcelona cpus will be "Family 10h"
  - Quadcore BKDG coming (available under NDA).
  - Some portions are useful to others than just Bios and Kernel developers. (e.g. Performance Counters)
- *Software Optimization Guide for AMD64 Processors*
  - Current version applicable through RevF processors.
  - Quadcore version publicly available.
  - http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/40546.pdf
- NERSC Technical Report *LBNL-62500*

  "Understanding and Mitigating Multicore Performance Issues on the AMD Opteron Architecture", John Levesque, Jeff Larkin, Martyn Foster, Joe Glenski, Garry Geissler (Cray Inc.), Brian Waldecker (AMD), Jonathan Carter, David Skinner, Helen He, John Shalf, Harvey Wasserman (LBNL/NERSC)

Trademark Attribution

AMD, the AMD Arrow logo and combinations thereof are trademarks of Advanced Micro Devices, Inc. in the United States and/or other jurisdictions. Other names used in this presentation are for identification purposes only and may be trademarks of their respective owners.